UNIVERSITY OF TORONTO SCARBOROUGH

Assignment: 1
Course: CSCC43
Deadline: Nov 24th, 11:59 pm
Submission: via Quercus

This assignment that makes up 5% of your total mark must be done individually. In this assignment you only implement a graphical user interface in Java for a home library (HL) system. The rest of the code will be completed in the third assignment.
With this application, you are able to insert and edit books and disks information that you have in your personal library. A disk can either contain one movie (a DVD) or a couple of music (a CD or vinyl). You are also able to view information about the books or disks that you have in your library or produce a few reports on your library. The detail of these reports will be explained in assignment 3.

In order to design the interface, you should know what data we store for each product. These data include:
Book:

- Name of the book, ISBN, publisher name, edition number, list of authors' name (surname, first name and middle name if applicable), number of pages, publication year, list of all the keywords by which this book is recognized relevant, and a description of the book
- Each book has at least one and at most 5 authors.
- The description of the book is at most 5000 characters long.
- The number of the keywords is at most 20 for each book.

Music Album:

- An album can have a few tracks. Therefore, the data that we need to store contains: Name of the disk (i.e. the album name or cover name), year published, data about each track that includes: the name of the music track, language of the lyrics, singer, song writer, composer, producer and arrangement.
- For simplicity, we assume that each sound track has only one composer, one arrangement, and one song writer.
- Each sound track can have at most two singers.
- Each album has one producer only.

Movie:

- Name of the movie, directors, script writers, cast, producers, composers, editors, costume designers and year of release. We are also interested to know the gender of the crew and whether or not they have received a well-known prize such as Oscar.
- We store at least one and at most three names for each role in the crew except for the cast for which we store at most 10 names.
- Each DVD contains one movie only.

The interface that you implement is an application that has a menu bar containing 3 options including *Data*, V*iew* and *Report*.

Data menu has three items; *Insert*, *Update* and *Remove*. Clicking on *Insert* should show you a sub-menu containing three items including Book, Music and Movie. Clicking on any of these three items should show a page in which users can enter all the corresponding attributes. At the bottom of the page there are two buttons, *submit* and *cancel*. If you submit, the data is stored in the tables while if you click on cancel nothing happens and data entry is ignored. In case *Update* is chosen, then a dialog box is shown in which a name is entered by users. This can be a name of a book, music album or movie. As soon as

ok is clicked, all the information about the product whose name was entered is shown. Users should be able to edit any of the attributes and then either save it or ignore it by clicking a corresponding button (i.e. submit, cancel). In case *Remove* is chosen, then again by entering the name of a product and confirming the its removal by pressing *ok*, the data is removed. If you cancel the removal you will click on *cancel* button.

- We assume that names are unique. So no book, movie or album has the same name.

Clicking on the *View* menu will show a page that lets you create a dynamic query. In this page, there are 3 check boxes including Book, Music Album and Movie and 2 text boxes. In the first text box you can either enter the name of the book, music album, movie or part of the name, and in the second text box you enter the year of publication or release. As soon as the *Submit* button is clicked, all the data that satisfies user input should be shown in a form of a 4 columns table that is titled Product's Name, Year, Type, Director/Singer/Author, which respectively shows:

- the full name of the product. i.e. the name of the book, music album, or movie
- the year that this product was released
- the type of the product, which is either a book (B), music disk (M) or movie (F).
- for this column, we are looking for ONE name only which is the name of the movie director, singer or book author. If more than one data is available for this column, the name that comes first alphabetically should be displayed. i.e. if the author name is Jane Smith and John Doe, then John Doe should be shown.

Clicking on *Report*, shows you a sub menu that has 10 items that for now is called R1, R2, …R10. The name of this ten reports will be given in your third assignment. Clicking on each of these reports lets your code to communicate with your database and make a query on it. The result should be shown in a form of a table, which is not editable.

This interface could be designed in a more aesthetic way, which requires more coding. Since the focus of this course is on database design, I preferred that you code less and concentrate on the database part more! I also recommend you to have a look at this tutorial https://docs.oracle.com/javase/tutorial/uiswing/components/menu.html in case you are not familiar with Java Swing.

Submission:

- You only submit one code, which is HL_YourUTORid.java (e.g. HL_abc123)
- When you complete the code for the third assignment, you should write a comment with the following format that shows which part of the code solves which question. You need to write this comment as close as possible to your SQL code. Here is the format:
  /* --- problem name --- */
  Examples:
  /* --- report 1 --- */
  /* --- Data->insert->book ---*/

Marking Scheme:

- I'd like to emphasize that this is not a Software Engineering course, therefore the way that you design this software has no mark, instead if it works perfectly with the third assignment and generates no problem, you will receive the full mark.

- Users can enter any invalid data as long as your code is resistant to SQL-Injection attack. You cannot accomplish this task now since you have to wait until the last week of the course when I teach the concept. If an invalid data is entered that does not compromise the security of the system, it should be accepted and no error should be generated. You will see later that the consequence of entering an invalid (and of course a secure) data is to receive an empty report or a DBMS error, which is fine for the purpose of this assignment together with the third assignment.
- Points will be deducted if your code crashes, or does not follow the layout that was proposed above.